Proceedings of Student-Faculty Research Day, CSIS, Pace University, May 5th, 2017

Driver Authentication based on Standard Onboard Diagnostics Data

Shreya Rajwade, Edison Castillo, Rushabh Pipada, Akshay Dikshit, and Anthony S. Richardson
Seidenberg School of CSIS, Pace University, New York, NY

Abstract— **A novel system was created to analyze and recognize driving styles. This system would then be utilized to differentiate divers just on the basis of their driving mechanism. This paper presents work in progress ideas which can be used for the development of a secured database to overcome the shortcomings of the current database system. This database would then be used as the master data source for the driver behavior analysis. This paper explores a MySQL based approach, to simplify the process of importing daily Comma Separated Values (CSV) files automatically, thereby allowing them to be easily accessible via Open Database Connectivity (ODBC) connection for analysis in an open data science platform powered by Orange.**

Index Terms— **Introduction, Background, Receiving data from OBD II, Web Application, Database, Conclusion, Future work**

I. INTRODUCTION

Driving styles have a major impact on vehicle and driver safety, fuel economy and other vehicles on the road. Due to a myriad of driving experiences, emotions and preferences, each driver has its own unique driving style or characteristic which can be observed based on data received from SAE/ISO standard Onboard Diagnostics (OBD) Data. Analyzing and understanding such driving styles can help control and maintain safe driving environments. Negative driving behavior like reckless driving poses a great threat all drivers, including the reckless driver. Some drivers do understand the ramifications of vulnerable driving; however, they still indulge in incompetent driving behavior. Knowing the driving style of individuals could be used to promote better driving and thus reduce road traffic accidents, especially if the negative driving incidents could be communicated to the driver in real-time or post-trip. Understanding driver styles will also improve fuel efficiency of a vehicle and thus reduce fuel consumption.

Recognizing driving behavior is a major challenge. However, with the advancement of technology, the International standards today mandate that all new vehicles must support an On Board Diagnostic (OBD) compliant interface. OBD is used in most of the cars and light trucks. [1]. Initially the OBD I was established and then a more sophisticated version i.e. OBD II was introduced. The OBD-II port can be found under the dashboard in the majority of current vehicles. It provides real-time access to a large number of vehicle status parameters. Furthermore, in case of malfunctions, Diagnostic Trouble Code (DTC) values are stored in the car ECU and can be later retrieved by maintenance technicians using proper tools [1]. OBD-II data can be read directly via a wired connection to the port and to a PC, or via a Bluetooth connection to a PC or Smartphone. This data can be downloaded and made available for further analysis via a secured database.

There are a few privacy concerns involved with the idea of tracking driver or vehicle. Misuse of this data might be possible which would create security concerns. The driver could be tracked based on vehicle and the data can be used for the wrong purpose. This would create a negative impact on the whole process of tracking driver and driver behavior.

This paper enhances the framework that would gather and analyze extensive data. It focuses on directly importing the a vast amount of CSV files to the MySQL database and thereby automating the entire process of extracting, transforming and loading (ETL) data. This paper also presents development of a web based application to display the number of files uploaded successfully and a query selection to track a driver by executing queries which would display the result of one or more drivers.

II. BACKGROUND

Driving behavior is believed to be unique to an individual. The way the driver turns a steering wheel in a vehicle depends on hand-eye coordination, hand shape and size, muscle control, foot strength and experience with the vehicle. Driving behavior is ubiquitous for automobile drivers so there is much motivation to be able to identify and verify a driver based on driving dynamics.

The goal of this project is to create a secure database to overcome the shortcomings of the current database system and to enable a researcher with a robust database source that can be used to analyze ~180 days of driving behavior data sets for ~300 drivers. The database intends to serve the purpose of master data source. This data is then used for further driving behavior analysis using Orange, an open source machine learning and visualization platform. The data will be recorded using the standard driving sensors equipped on all the vehicles in the study. [2] The data is then stored in .CSV format on web server maintained by the vendor who is aggregating the data. Each data file is recorded with a unique ID, which acts as a primary key. However, this ID represents the vehicle and not the driver per se. The vendor emails a scheduled daily email blast to the research, containing a zip file of all the driving data from the prior day for all ~300 drivers. All these C0053V files are then accumulated and collectively form a single day's data. Another goal of the project is to implement a web application to display the number of files imported in the database on a daily basis. It also involves implementation of user executing queries to track a particular driver and analyze his or her driving behavior.

Currently, these files are being combined using MS Access and .BAT format. This project uses an enormous amount of data files, totaling over 30GB. Due to the limitation of MS Access, that database can only allow 2GB of data to be manipulated. Thereby, allowing only data of approximately 1 day (approx. 1GB of data) to be maneuvered and manipulated. This leads to a significant delay in further research of the driving dynamics gathered.

Various difficulties may arise while implementing the database import like the presence of null values in the files and how to store these null values. There might also arise difficulties while importing the data with concern to processing time as the user is importing huge data (up to 1.5 GB) on a daily basis in the database. This would thus lead to a delay in importing the files conveniently in the database. Apart from these difficulties, there are user considerations to take in account, such as if a failure occurs while importing a particular file in the database. Concerns regarding duplication of data and measures to get rid of the duplicate data in the database might also occur. Importing 1GB of data on average per day can also present a problem because the database can grow rapidly and become a problem depending on where the database will be hosted. This system should be updated to use a big data database (i.e. Hadoop File System) system no later than May 2018, to avoid any issues related to the database size, like performance.

III. RECEIVING DATA FROM OBD II

The web based application intends to communicate with the vehicle using the OBD II parameter IDs (PID)[3]. The parameter IDs provide information about the vehicle like speed, fuel consumption etc. The information which is received is stored in the app. This app then imports data to the SQL database.

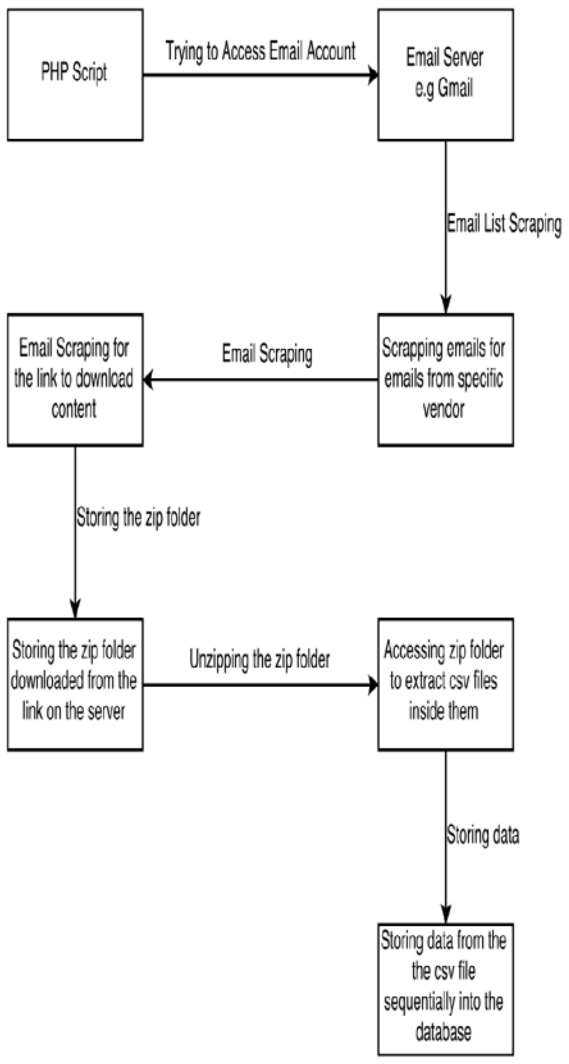

PHP Script
Trying to Access Email Account
Email Server e.g Gmail
Email List Scraping
Scrapping emails for emails from specific vendor
Email Scraping
Email Scraping for the link to download content
Storing the zip folder
Storing the zip folder downloaded from the link on the server
Unzipping the zip folder
Accessing zip folder to extract csv files inside them
Storing data
Storing data from the the csv file sequentially into the database

Figure 1: The Data Flow Diagram of automated data extraction

This process can be done manually by uploading individual CSV files, however we are creating an automated script to handle this process in the backend without any user interference.

Automated Data Extraction tries to access the Emails received from the vendor and extract the data from the emails. This process can be handled in a six stage script, each performing small jobs to handle the overall automation.
Stage 1: A PHP script will access the Email server on which will receive emails from the vendor using the IMAP function where the script accesses the inbox of the required email address.
Stage 2: The PHP script will then look for emails from the

vendor who is sending the OBD data, in the inbox and filter them out.
Stage 3: By selecting the most recent email from the vendor, the script "scrapes" the email server to find the link which downloads a zip file containing the data.
Stage 4: The script downloads the zip folder to the server in a directory, from the link in the emails
Stage 5: Since there is a requirement to house the folder on the server in the directory, the script will need to access the zip folder to extract the CSV file inside the directory.
Stage 6: The most important part is to save the data from the CSV file into the database, which is achieved in this step. This occurs by first arranging the files in the directory to access the most recent file. Then, the script accesses the content of the CSV file, and matches the column header of the CSV file to the column name in the database table and then sequentially imports each row of data into the database on which will be used to run the queries and connect to Orange, an open source machine learning and visualization platform.

Automated data extraction without user interference reduces the likelihood for user injected error which might have been introduced by user interference.

IV. WEB APPLICATION

A web application is necessary to act as an interface and update the user with the summary of the data imported to the database by automated data extraction. The application comprises of total 5 pages: 1) a login page for checking the authenticity of the registered user, 2) a sign up page to register new user, 3) an interactive page for changing the account settings, 4) an import page to allow users to browse for a file and import it to the database, and lastly 5) a home page where the summary of the last import will be displayed along with some other useful information. This web application is responsive and users will be able to use it from their mobile.

A. FRONT END IMPLEMENTATION

The application is based on three technologies - HyperText Markup Language version 5 (HTML5), Cascading Style Sheets 3 (CSS3) and JavaScript.

HTML is used as the markup language for creating the page layout of the application. HTML elements were used to construct the objects, interactive forms and display images in the application.

HTML only provides a figurative and illustrative layout thus Javascript is needed for validation and changing the behavior and the content of the pages. In order to make the application visually appealing to the user, CSS3 is being used to design and define the look and the layout of the content.

Figure 2. below shows the SIGN IN page, it is the front page of the application. The user has to input his / her credentials to access the data. Like any other application, if the user does not have an existing account for the application, he / she will have to register and create an account. The link to the registration page is given on the SIGN IN page.

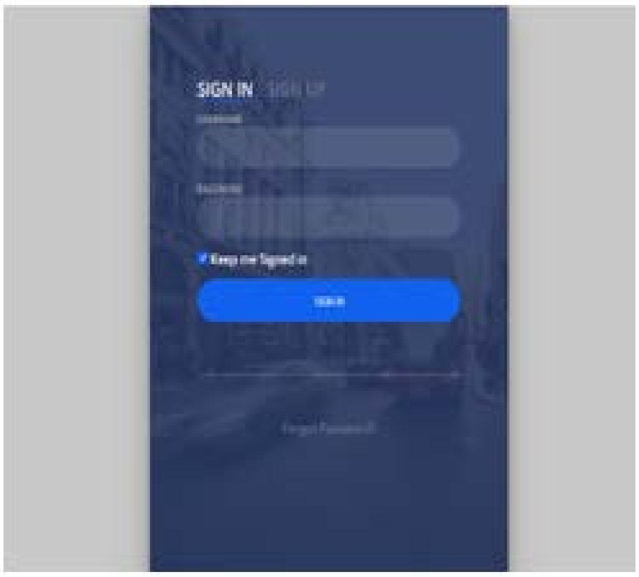


Figure 2. Sign-in Page

Figure 3 below displays the home page where it provides the link for viewing CSV file by different categories. The link to edit profile, link to logout and link to go to the Import page.

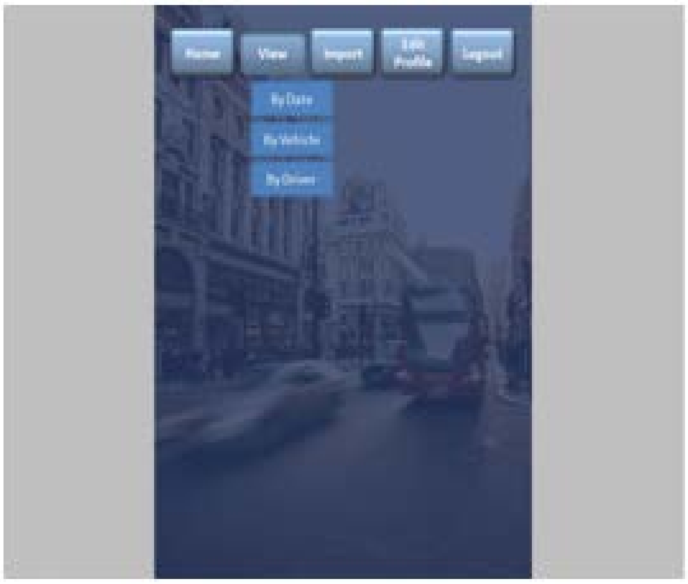


Figure 3. Main Page

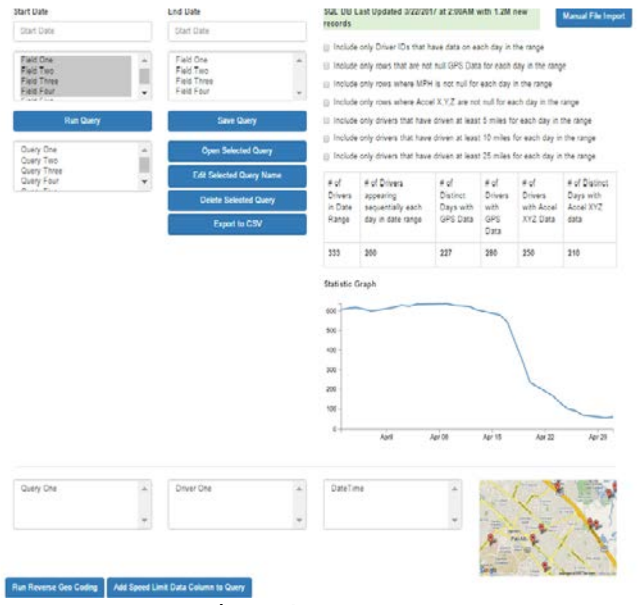
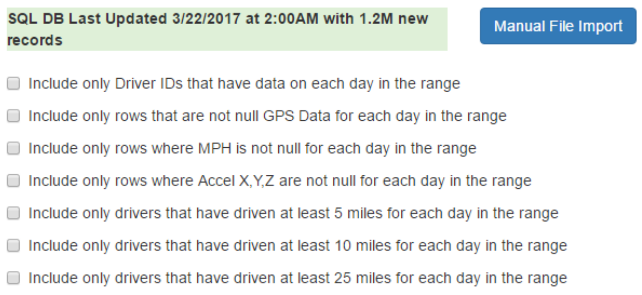


Figure 4. Home Page

Figure 4 displays the Home Page of the system. It provides the user with the number of files and the size of files imported to the database. It also provides information if there were any errors while uploading a particular file. It allows the user to enter Start Date and End Date of the data the user wants to be displayed. It also gives user the option to display the number of fields the user wants. The Home Page displays various options for user in terms of working on the queries. It gives the user the option to select a particular query, edit it, delete it etc.

Before running the query, the user is provided with a choice to include only a particular field data in the table to be displayed. This option helps user to get rid of the large number of fields which makes it difficult to analyze the data. Thus, it enables prompting of the minimum required fields. Figure 5 below displays the option to display only the required fields.


SQL DB Last Updated 3/22/2017 at 2:00AM with 1.2M new records

Manual File Import

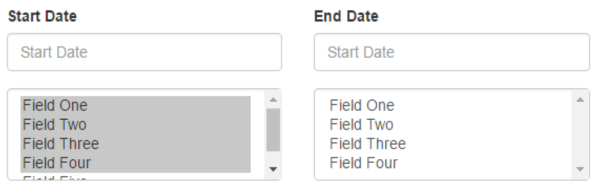
- ☐ Include only Driver IDs that have data on each day in the range
- ☐ Include only rows that are not null GPS Data for each day in the range
- ☐ Include only rows where MPH is not null for each day in the range
- ☐ Include only rows where Accel X,Y,Z are not null for each day in the range
- ☐ Include only drivers that have driven at least 5 miles for each day in the range
- ☐ Include only drivers that have driven at least 10 miles for each day in the range
- ☐ Include only drivers that have driven at least 25 miles for each day in the range

Figure 5. Query Dynamic Criteria

The user can customize queries by selecting one or multiple criteria from a list of predefined criteria. The user is provided with an option to save query which they has just implemented. Once the desired query is built, the saved query can later on be retrieved and customized.

Figure 7 below displays the option to run the query after creating it. It also displays a save option provided to the user to save the query so that it can later be retrieved and customized.

Query Field selection will allow user to select which field they want in the query. For example, of fields A, B, C and D the user can select only B and C to be displayed in the query. Figure 6 below display the Query field selection option for the user.


Start Date

Start Date

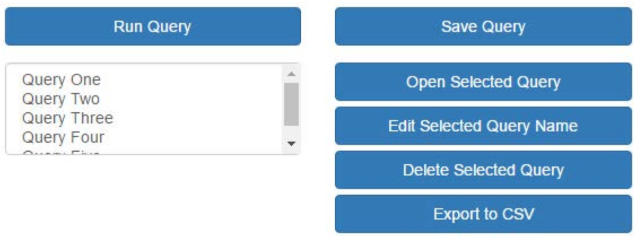
Field One
Field Two
Field Three
Field Four

End Date

Start Date

Field One
Field Two
Field Three
Field Four

Figure 6. Query Field Selection


Run Query

Query One
Query Two
Query Three
Query Four

Save Query

Open Selected Query

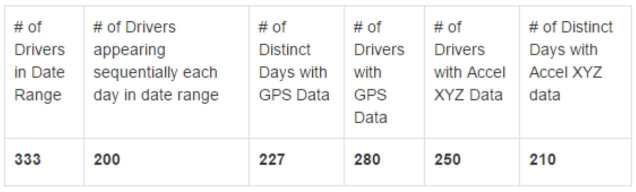
Edit Selected Query Name

Delete Selected Query

Export to CSV

Figure 7. Run and Save Query

Once the query is executed, a table with statistics about the retrieved data will be populated. Figure 8 below displays such statics of the populated data.



# of Drivers in Date Range	# of Drivers appearing sequentially each day in date range	# of Distinct Days with GPS Data	# of Drivers with GPS Data	# of Drivers with Accel XYZ Data	# of Distinct Days with Accel XYZ data
333	200	227	280	250	210

Figure 8. Populated data

The user is provided with a choice to analyze and display data in the form of histogram. The user could select a date range and factors they want the histogram to be based on. The histogram is then displayed to the user to give a clear analysis of the data to be displayed. The graph can be used to analyze the maximum requirements of vehicles on a particular day etc.

Apart from the above mentioned choices, the user is also provided with an option to carry out geo location to track a particular driver.

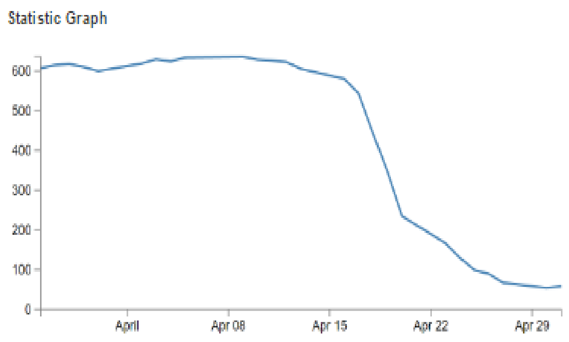


Figure 9. Histogram

Figure 9 represents driver details on a particular day. For example, it can display driver requirement on a particular day. It can enable the user to know which day there is maximum requirement of drivers, for example on holidays or weekends. The application will display a histogram about the driver's data being logged. This histogram could be achieved using different client side libraries. The one proposed to be used is the d3js library. This library is very powerful and relies on the most advanced and latest technologies. It uses HTML, SVG and CSS to draw its many type of graphs and documents.

Tracking of driver location to road is another feature of this project. By building a reverse geocoding web service, the user can pinpoint the driver's location to a road and associated speed limit for that road. The sample files already contain GPS data on most of the drivers. Reverse Geocoding is the process of finding an address, or another type of resource for a given lat/lng pair. GeoNames offers a wide range of reverse geocoding web services.

The Google Maps APIs web services are a collection of HTTP interfaces to Google services providing geographic data for map applications. The Google Maps Roads API identifies the roads a vehicle was travelling along and provides additional metadata about those roads, such as speed limits. The Google Maps Roads API allows mapping of GPS coordinates to the geometry of the road, and to determine the speed limit along those road segments. The API is available via a simple HTTPS interface, and exposes various services like Snap to Roads, Nearest Roads, Speed Limits. The Google Maps Roads API returns the posted speed limit for a given road segment. In the case of road segments with variable speed limits, the default speed limit for the segment is returned.

The application will provide the user a way to analyze a route that a driver took. Information such as speed limit on each of the road will be displayed.

Figure 10 displays the working of reverse geocoding. The user could select a query from the section and then select the driver's ID whom he wants to track. The user can select the day for which he wants to track the driver data (speed limit, distance traveled etc.).

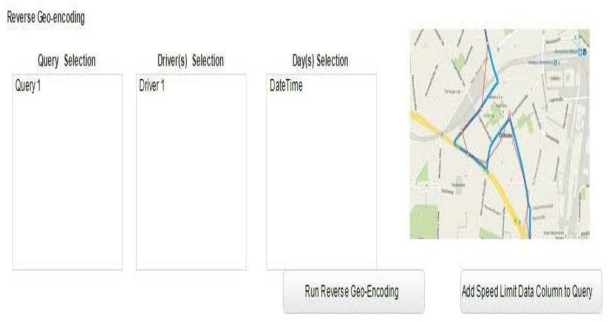


Figure 10. Reverse Geocoding

B. DATABASE

Such high volume of data definitely poses an obstacle in the process of analyzing. There are many software and toolkits that can be utilized for the purpose of storing big data. However, this paper uses MySQL as the data source. MySQL started out as open data base management software. It enables backend data to be stored in large quantities, executes queries very fast and uses a thread based memory allocation system. [4]

For the purpose of storing driver behavior analysis data MySQL version 5.5 is being proposed in this paper. The daily CSV files gathered through the web application will be uploaded directly into the database. These data files will be aggregated and organized for future use. The files will be organized according to the unique vehicle ID and will be tracked on these IDs. The files contain columns of data comprising of different parameters. The SQL database plans to store 800 MB – 1.5 GB of data daily. If 1 GB of data on average gets transferred to the database and the database is hosted on a Linux server, this application will remain reliable for at least two years. This will give the future research teams enough time to enhance the system to work with a database system that supports massive data processing and storage.

The database design does not only support the import of the driver behaviors analysis data, but also logins. This application needs to protect the data being imported and for that only authorized users will be able to login. This database architecture will also support import history and status and custom queries. The application will be able to pull data of drivers that have only been active for a specific period of time and to show sets of data where there are no null columns or values.

Figure 11 below displays the database design. The database design is composed of 5 entities. Two of these entities are for supporting the data import for each driver. The other three entities are for allowing only authorized users to look at the history of import and status. Each entity has its primary key which will reduce redundancy.

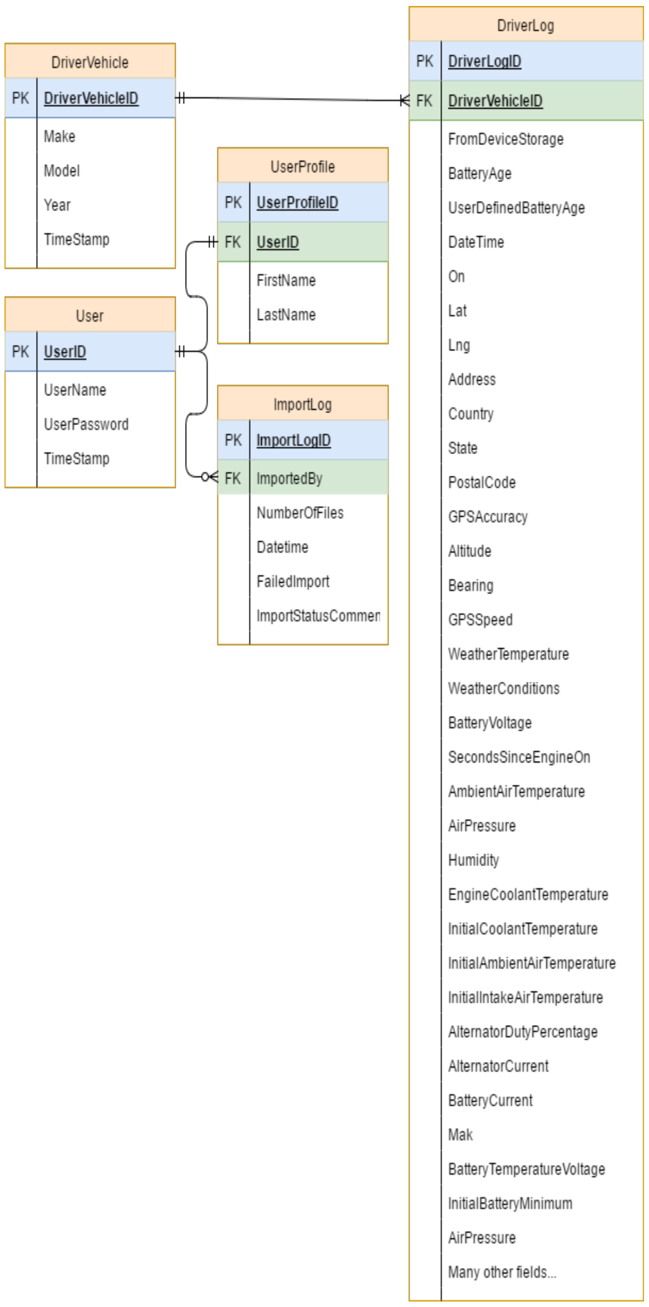


Figure 11. Database Design

V. CONCLUSION

The proposed solution of using MySQL as a primary data source would be a pragmatic and economical approach to the data storing issue. It will provide an effective measure to study and analyze large amount of data. It would also help track drivers and their driving behavior to establish a particular standard. This solution will aid in developing analysis of ~300 drivers to determine which drivers meets the standard to be categorized as a safe driver. It would thus aid researchers in determining methods for reduction of road traffic accidents, and promote maintain safe driving behavior, based upon deep analysis of the driving styles and environments. This application will enable the research on driving behavior to continue without any obstacles. It will help categorize driver behavior in various categories as the analysis of the data would become less difficult. Thus, establishing a method to evaluate unsafe driving. The web application would also enable users to identify the number of files imported to the database, the amount of data imported and also if any failure occurred while importing data through CSV files to the database.

VI. FUTURE WORK

The proposed database and the web application are in final testing and will be released to the customer to perform analysis during the summer of 2017. The is an opportunity to consider other options to make the effort less tedious such as an internal process that runs multiple times a day.

Careful consideration in the development of the logic for login to the application is required. This logic is critical because protecting driver's data is a top priority of the customer. The objective is to have a fully functional login process that will prevent unauthorized users from accessing the application.

Currently there is significant historical data available that needs to be migrated, organized and aggregated in the new database. There is an effort underway to collect and import the data into the new database for future use. Once imported, the initial requested queries will be developed to allow for analysis of the data to make sure that the database will support the data that will be coming in from the CSV files.

Once the main core of the import process is completed, the UI will be enhanced. Bootstrap 3 will be used; it is one of the most popular front-end frameworks available right now. Bootstrap will allow the developers to create a functional, user friendly and responsive layout that will be accessible from almost any device that support Chrome, Firefox or Safari.

The development team is also working on automating the process to a much simpler level for the user. Whereby users will not need to browse or import the files. Once imported, the process will notify the users with a status of the import. The tracking of driver's driving behavior has improvement or deterioration over a period of time through a graph can also be implemented. The team is also working to address the security concerns related to the data.

Lastly, the team will connect the MySQL database to the data mining platform (Orange). The ultimate goal of this importing process and the application is to be able to analyze all the imported data in Orange. Once in Orange, users will be able to further explore the possibilities of authenticating drivers based on their driving behaviors.

REFERENCES

[1] Ruta, M., Scioscia, F., Gramegna, F., Di Sciascio, E.: A Mobile Knowledge-Based System for On-Board Diagnostics and Car Driving Assistance. In: The Fourth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM2010), IARIA (2010) 91–9

[2] Malintha Amarasinghe, Sasikala Kottegoda, Asiri Liyana Arachchi, Shashika Muramudalige, H. M. N. Dilum Bandara, Afkham Azeez: Cloud-Based Driver Monitoring and Vehicle Diagnostic with OBD2 Telematics. In: 2015 International Conference on Advances in ICT for Emerging Regions (ICTer)

[3] Lin, C.E., Shiao, Y.S., Li, C.C., Yang, S.H., Lin, S.H., Lin, C.Y.: Real-Time Remote Onboard Diagnostics Using Embedded GPRS Surveillance Technology. Vehicular Technology, IEEE Transactions on 56(3) (2007) 1108–1118S. Zhang, C. Zhu, J. K. O. Sin, and P. K. T. Mok, "A novel ultrathin elevated channel low-temperature poly-Si TFT," *IEEE Electron Device Lett.*, vol. 20, pp. 569–571, Nov. 1999.

[4] Lee J. (2013, November 30). Oracle vs. MySQL vs. SQL Server: A Comparison of Popular RDBMS [Online]. Available: https://blog.udemy.com/oracle-vs-mysql-vs-sql-server/R. E. Sorace, V. S. Reinhardt, and S. A. Vaughn, "High-speed digital-to-RF converter," U.S. Patent 5 668 842, Sept. 16, 1997.

[5] [5]https://developers.google.com/maps/documentation/roads/intro

[6] http://www.geonames.org/export/reverse-geocoding.html

[7] https://d3js.org/